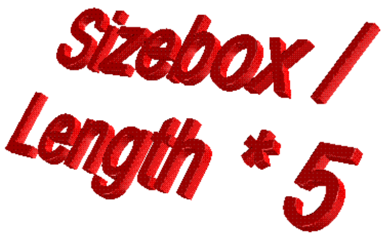
ž

ž

ž

CHAPTER 4 ● Formulas

IntelliShapes in TriSpectives gain their intelligence through a variety of mechanisms, one of which is the ability to represent a shape’s parameters as formulas. These formulas relate a value to one or more other values.



In this chapter

- Introduction to formulas
- Formulas and property sheets
- Formula example
- Formula reference

Ž
Ž
Ž

● Introduction to formulas

The relationships created by assigning formulas rather than fixed values to an IntelliShape’s parameters result in dynamic shapes that respond to actions other than direct manipulations by the user.

When you drag a block out of the catalog and drop it in a scene, its length, width and height are fixed values. If you pull on its handles, you change the values and the block retains the new parameters. You could also open the IntelliShape Properties sheet for the block, select the Sizebox tab and enter new values for the block’s width, length or height. In both cases, fixed values define the block’s extents. It will retain its size until you take another one of the above actions to change it.

Now open the IntelliShape Properties sheet for the block, select the Sizebox tab, but this time check the Show Formulas box in the lower left. You should see, in this case, the same values you saw before you checked the Show Formulas box. However, the system is now expecting you to input a formula for the selected parameter rather than a fixed value. If you enter “Length” in the Width edit box and “Length” in the Height edit box, then the width, length and height of the block will be equal upon acceptance of the property sheet change. Furthermore, if you now pull the block’s length handle, the width and height of the block will follow the length. The block maintains itself as a cube with its extents driven by the length.

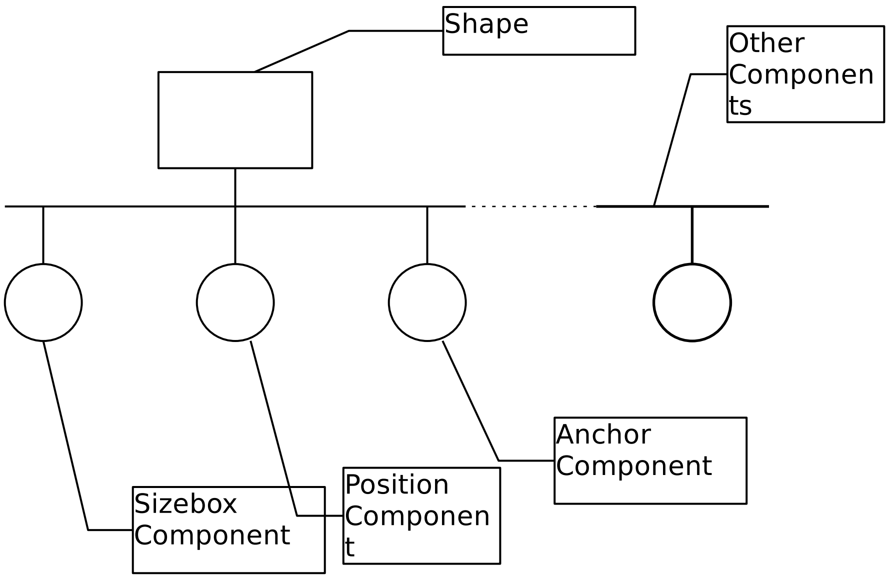
Note that a pull or push on the width or height handles will replace the respective formulas on those extents with the new values resulting from the handle action. In general, the formula on a shape parameter is replaced when a user action generates a new value directly for that parameter.

Formulas are used extensively within TriSpectives to preserve geometric relationships. For example, the vertices of 2D profiles forming the basis of extrude, spin, sweep and loft shapes are represented by formulas, rather than assigned spatial values. These formulas are typically a normalized value (a number between 0 and 1) multiplied by the sizebox width or length. With the vertices represented in this manner, the 2D profile of the shape stretches or shrinks as the sizebox is changed. Much of the intrinsic behavior of shapes and dimensions in TriSpectives is achieved using formulas attached to shape parameters. Users are given access to most of these parameters via property sheets and OLE Automation, enabling them to build IntelliShapes of their own design.

In order to write effective formulas for shapes, you have to know a bit about how shapes are organized, both internally and in relation to other shapes in your scene or on your page. If, for example, you want to make the position of shape A relative to the position of shape B, you have to know how to refer to B's position *from* A's position. Such a reference is not absolute, but depends upon how A and B were generated. This dependency is not a function of the spatial relationship between the two shapes, but of the underlying software architecture connecting all shapes in your scene. This is not as esoteric as it might sound. If you can picture the organization of directories and files on your computer, perhaps as it appears in your Windows95 Explorer, then you can understand the organization of TriSpectives shapes and their components.

Most file systems organize disk drives into a hierarchy of directories terminating at the "root" directory. Directories contain both files and other directories. We call the overall shape of this structure a tree (it is upside down, with the root at the top). Most of us navigate this tree regularly by now, and we know that the path from one point in the tree to another involves either going down the tree, up the tree, or up the tree and back down another branch. TriSpectives also organizes and references shapes in this manner.

A single shape is analogous to a directory; it actually contains very little information. It does contain components, analogous to files, which contain the information characterizing the shape. All shapes have at least one component: the Position component. As its name implies, the Position component supplies the shape's spatial coordinates and orientation. The Sizebox component defines the spatial extents of the shape. The



Anchor component defines a point on the shape where it attaches to other shapes. Overall, more than 30 different components can plug into shapes.

Figure 1: Shape and components

Shape components contain the variables to which the program or the user assigns values or formulas. If you display the IntelliShape properties for a given shape, and select the Sizebox page, you will see edit boxes for Length, Width and Height. When entering a formula into a component's property page, you are "in" that component and can therefore reference its variables directly. This is why, in the example above, we need only enter "Length" in the Width edit box to reference the sizebox Length. If we want to make the sizebox

Width a function of the shape’s position, then we have to reference a variable in the Position component.

The syntax to reference the X coordinate of a shape's position from another component of the same shape is:

Position\X

The syntax to reference the sizebox length from another component of the same shape is:

Sizebox\Length

Note that these names can differ in localized versions of TriSpectives. TriSpectives converts and stores user or Automation entered formulas into a locale independent format.

As you build your model, TriSpectives is building the hierarchy of shapes. Each shape that drops in becomes the child of either the "root" shape or the shape that it dropped on. You can see these relationships graphically using the TriSpectives WorkBook Browser. To create formulas which reference across shapes, we extend the syntax introduced above to traverse the parent-child relationships of the shapes.

Figure 2: Sample shape hierarchy

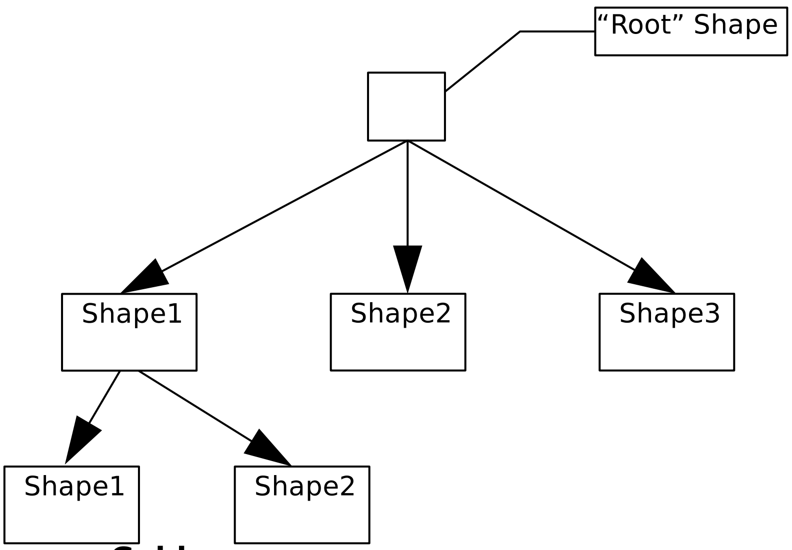


Figure 2 shows a possible shape hierarchy. The names Shape 1, Shape 2, up to Shape n, are assigned by the system as the shapes are created. The names are called the system names and are shown in the General tab of the shape's property sheet. These names are used when referencing from shape to shape. Most of the time these references will be through the parent shape to some component of a sibling shape. For example, using Figure 2 as a reference, we can assign a formula to the position of Shape 1 (the child of the root) to make it always be offset from Shape 2 in X. We assign this formula to the X variable of the Position Component of Shape 1:

Parent\Shape2\Position\X + 6 in

to give a 6 inch X offset between the shapes. Note that from Shape 1 we get to Shape 2 through their common parent. If you move Shape 2 at this point, Shape 1 will follow. If you move Shape 1, the formula in the Shape 1 Position Component is overwritten with the new Position value.

The following rules apply:

- All constants appearing in formulas are assumed to be in system units; centimeters for length, radians for angles. To reference a constant as another unit, append a conversion intrinsic to the constant. In the example above, the X offset is specified to be in inches. Regardless of the current user units in the system, this offset will always be 6 inches. If "in" was not appended to "6", the offset would default to 6 centimeters, regardless of the current user units.
- The system creates shape names by appending a number to the word Shape, as in Shape1 and Shape2. This is the Shape ID.
- The Shape ID is unique for all sibling shapes (children of the same parent). However, Shape ID's may be repeated at different levels of the tree without causing ambiguity (see Figure 2).
- Shape ID's start at 1 and increment as shapes are added by insertion or drag and drop.

- Deleting a shape does not reset the shape ID. After deletion, Shape ID’s do not decrement or skip or fill in unused values.
- The keyword *Parent* traverses up the tree, the Shape Name, *ShapeID*, traverses down the tree.

Formulas and property sheets

You can view and edit formulas or values via property sheets and dialogs throughout TriSpectives. Table 1 shows the behavior of the edit boxes within property sheets and dialogs.

Show Formulas	Existing Formula?	User Input	Resultant Action
ON	NO	value	Assign value
ON	NO	formula	Assign formula
ON	YES	value	Assign value
ON	YES	formula	Assign formula
OFF	NO	value	Assign value
OFF	NO	formula	Calculate value from formula. Assign value.
OFF	YES	value	Prompt before assigning value.
OFF	YES	formula	Calculate value from formula. Prompt before assigning value.

Table 1 : Property sheet operation

Note that some modes simply calculate the value from the entered formula, and assign the result to the parameter. This is known as *calculator mode*, and it allows you to do simple math operations right in the edit box. For example, if you want to enter a Sizebox width as 4 and 1/32 inches, you could enter 4.03125 or you could enter 4 + 1 / 32. The latter is determined to be a calculation, which TriSpectives executes and assigns to the Sizebox width. References to system variables and conversion intrinsics are not allowed in calculator mode.

However, intrinsic functions such as SIN, COS, etc, are allowed. Note that constants in calculator mode are in user units, but constants entered when the Show Formulas check mark is set are in system units, centimeters for length and radians for angles.

Because of the above rule, TriSpectives provides conversion intrinsics which give you a way to specify units for constants in a formula. Table 2 is a list of these intrinsics.

Unit Name	Converts Constant To:	Sample Usage (when Show Formulas is ON)
in	inches	Sizebox\Length + 1.5in
ft	feet	Parent\Shape6\Position\X * 2 + 100 ft
mi	US Statute Miles	5280 ft + 10 mi
mm	millimeters	Sizebox\Width / 10.05 mm
cm	centemeters	Width + 50 cm + Length * 2
m	meters	Width - 0.5e-6 m
deg	degrees	PI ()/ 180 deg
rad	radians	10 rad

Table 2: Intrinsic conversion functions

● Formula example

Formulas are pervasive within TriSpectives, but that doesn't mean you have to care. Every shape you drag from the catalogs or create with the tools is defined by formulas, and you can see this by bringing up the IntelliShape property sheet, and selecting the Outlines page from the Cross-Section sheet (press Properties). This page shows you the formulas at the core of the shape. Arbitrarily messing with these formulas is not recommended, but if you have an idea of what's going, you can't really hurt anything (save your work, first!).

There are some shapes that make more advanced use of formulas, these can be found in the AdvShapes catalog (among others). Drag out the Pipe shape from this catalog and drop it on a scene. View this shape's Outlines page (as described above). The pipe is extruded from two circles, one representing the outer diameter (outline 1) and one the inner diameter (outline 2).

The third column (3) of the outline for a circle is the radius. Note that for this shape the radius of the circle in Outline 1 is the formula:

```
Parent\Variable\Number[0]
```

And the radius for the circle in Outline 2 is the formula:

```
(Parent\Variable\Number[0] - Parent\Variable\Number[1])
```

Now cancel the Cross-Section property sheet (keeping the IntelliShape property sheet in front of you). Select the tab marked Variables. These are the variables referred to by Parent\Variable in the Outlines (Variable is a component of the Outlines' parent shape). Number[0] and Number[1] refer to the values in column

2 of row 1 and row 2, respectively. Variable\Number[0] is labeled Outside Radius and Variable\Number[1] is labeled Wall Thickness.

By typing in values for Wall Thickness and Outside Radius, you can watch these properties of the pipe change. This is because the outlines from which the solid is extruded are connected to these values via the formulas shown above. The radius of the circle in Outline 1 is directly connected to the Outside Radius value, and the radius of the circle in Outline 2 is the difference between the Outside Radius and the Wall Thickness.

Ž
Ž
Ž

Formula reference

You can write formulas using standard mathematical operators: *, /, +, -. In addition, a library of intrinsic functions provides support for conditional if, min, max, trigonometric functions and others. The intrinsic functions are listed at the end of this chapter.

There are also intrinsic operators for unit conversion. These are listed in Table 1, above, along with sample usage.

The following sections describe the available functions.

Function Name: PI

Syntax: PI

Return Value:

PI returns the constant pi: 3.14159...

Function Name: ABS

Syntax: ABS(X)

Return Value:

ABS returns the absolute value of X.

Function Name: SIN

Syntax: SIN(X)

Return Value:

SIN returns the sine of X, where X is an angle in radians.

Function Name: COS

Syntax: COS(X)

Return Value:

COS returns the cosine of X, where X is an angle in radians.

Function Name: TAN

Syntax: TAN(X)

Return Value:

TAN returns the tangent of X, where X is an angle in radians.

Function Name: ASIN

Syntax: ASIN(X)

Return Value: float

Function Name: ACOS

Syntax: ACOS(X)

Return Value: float

Function Name: ATAN

Syntax: ATAN(X)

Return Value:

ATAN returns the arctangent of x. If x is 0, ATAN returns 0. ATAN returns a value in the range -pi/2 to pi/2 radians.

Function Name: ATAN2

Syntax: ATAN2(Y, X)

Return Value:

ATAN2 returns the arctangent of y/x. If both parameters of are 0, the function returns 0. ATAN2 returns a value in the range -pi to pi radians, using the signs of both parameters to determine the quadrant of the return value.

Function Name: SQRT

Syntax: SQRT(X)

Return Value:

SQRT returns the square root of X. If X < 0, the evaluation fails and the value of the variable is unchanged.

Function Name:	IF
Syntax:	IF(X, Y, Z)
Return Value:	
	IF evaluates X. If X evaluates to TRUE (non-zero), Y is returned, else Z is returned.

Function Name:	MAX
Syntax:	MAX(X, Y, ..., Z)
Return Value:	
	MAX evaluates all arguments and returns the largest value. MAX takes a variable number of arguments.

Function Name:	MIN
Syntax:	MIN(X, Y, ..., Z)
Return Value:	
	MIN evaluates all arguments and returns the smallest value. MIN takes a variable number of arguments.

TriSpectives Variables

Legend:

Data:	type	L: length (shown in centimeters)
		A: angle (shown in radians)
		T: scalar (dimensionless)
		N: integer
		S: string
	(a-b)	a: lower limit, b: upper limit
	[d]	d: default value(s)

SC - Shape Components

Name	Data	Description
Parent		Contains information to go upward in the shape hierarchy
Shape		Contains information to go downward in the shape hierarchy
SizeBox		Contains parameters to control the size of a shape along its length, width, and height
Path		Contains a simply-connected set of 2D curves
Anchor		Contains the transformation from the corner of a shape’s size box to its drop position
Position		Contains the transformation from the corner of a shape’s parent’s size box to its drop position
Group		Contains a group of shapes
AttachPt		Contains the definition of an attachment point
Facet		Contains a facet model (models defined by polygons only)
Brep		Contains a model in boundary representation

Sweep		Contains a model created by sweeping a profile - extrude, turn, and general sweep
Loft		Contains a free-form model created by interpolating a set of cross-sections
Compound		Contains a model created by combining one or more primitive and/or compound shapes
Profile		Contains an indirect reference to a path
EdgeBevel		Contains information of blends or chamfers on one or more edges
TriBall		Contains the location of a triball
SolidInstance		Contains a reference to a Brep, Sweep, Loft, or Compound
Text		Contains text data
UiBehavior		Contains control information of what happens when a shape is dragged, dropped, or double-clicked
PathSweep		Contains an indirect reference to a path, for general sweep and loft
Light		Contains a light for illuminating a scene
ProfileGroup		Contains a group of profiles (for lofting)
LoftSection		Contains the position and orientation of a cross-section used for lofting
Dimension		Contains parameters to measure and to anotate
Variable		Contains data of an array of variable values and names
SmartDimension		Contains parameters to position a shape

Null		Contains nothing, used as a sentinel value
ParentInstance		Contains information to go up in the shape hierarchy
UserHandle		Contains parameters to position user-defined handles
EmbeddedPage		Contains data of an embedded page

SIZEBOX - Size Box

Name	Data	Description
Length	L	Length of sizing box
Width	L	Width of sizing box
Height	L	Height of sizing box
HideLengthHandles	N(0-1)	Suppresses display of length handles to prevent interactive change 0: do not suppress handle display 1: suppress handle display
HideWidthHandles	N(0-1)	Suppresses display of width handles to prevent interactive change 0: do not suppress handle display 1: suppress handle display
HideHeightHandles	N(0-1)	Suppresses display of height handles to prevent interactive change 0: do not suppress handle display 1: suppress handle display
LinkHandles	N(0-4)	Sets various aspect locks for use with the handles 0: no link 1: length is linked to width

		2: width is linked to height 3: height is linked to length 4: length, width, and height are all linked
LengthFixedPt	N(0-2) [0]	Specifies the point that remains stationary when a length handle is pulled 0: anchor point 1: opposite handle 2: mid-point between handles
WidthFixedPt	N(0-2) [0]	Specifies the point that remains stationary when a width handle is pulled 0: anchor point 1: opposite handle 2: mid-point between handles
HeightFixedPt	N(0-2) [1]	Specifies the point that remains stationary when a height handle is pulled 0: anchor point 1: opposite handle 2: mid-point between handles
2DDiagFixedPt	N(0-2) [0]	Specifies the stationary point when a handle along the diagonal through the origin is pulled (2D only) 0: anchor point 1: opposite handle 2: mid-point between handles
Type	N(0-1)	Type of box 0: 3D 1: 2D
DisplayType	N(0-3)	Display style of sizing box 0: both box and handles 1: box only 2: handles

		3: no display
FixAnchor	N(0-1)	Behavior of anchor point during resizing 0: anchor point may move in space 1: anchor point will not move in space

XFORM - transformation

Name	Data	Description
X	L[0]	Translation along x-axis
Y	L[0]	Translation along y-axis
Z	L[0]	Translation along z-axis
RotX	T[0]	X-component of rotation axis
RotY	T[0]	Y-component of rotation axis
RotZ	T[1]	Z-component of rotation axis
RotAngle	A[0]	Rotation angle

PATH - a simply-connected set of curves

Name	Data	Description
X	L ARRAY	X-coordinate of end-point of a curve (the start-point is defined by the previous curve)
Y	L ARRAY	Y-coordinate of end-point of a curve (the start-point is defined by the previous curve)
lengthA	L ARRAY	The meaning of this variable depends on the curve type: Start: not applicable Line: not applicable Ellipse: radius of primary axis

		Elliptical arc: a fraction with a rather difficult definition; draw a line between the end points, call this the chord; draw the perpendicular bisector of the chord to intersect the arc; the denominator of the fraction is the chordal length; the numerator is the distance between the bisection point and the intersection point at the arc. Bezier spline: x-component of tangent at start-point Fillet: radius
lengthB	L ARRAY	The meaning of this variable depends on the curve type: Start: not applicable Line: not applicable Ellipse: radius of secondary axis Elliptical arc:not applicable Bezier spline: y-component of tangent at start-point Fillet: not applicable
lengthC	L ARRAY	The meaning of this variable depends on the curve type: Start: not applicable Line: not applicable Ellipse: not applicable Elliptical arc: not applicable Bezier spline: x-component of tangent at end-point Fillet: not applicable
lengthD	L ARRAY	The meaning of this variable depends on the curve type: Start: not applicable Line: not applicable Ellipse: not applicable

		Elliptical arc: not applicable Bezier spline: y-component of tangent at end-point Fillet: not applicable
Angle	A ARRAY	The meaning of this variable depends on the curve type: Start: not applicable Line: not applicable Ellipse: not applicable Elliptical arc: angle as measured from the x-axis to the primary axis Bezier spline: not applicable Fillet: not applicable
ScalarA	T ARRAY	The meaning of this variable depends on the curve type: Start: 0: curve is open 1: curve is closed Line: starting trim parameter (for filleting) Ellipse: eccentricity (fraction, the length of the secondary axis over the length of the primary axis) Elliptical arc: eccentricity (fraction, the length of secondary axis over the length of the primary axis) Bezier spline: not applicable Fillet: not applicable
ScalarB	T ARRAY	The meaning of this variable depends on the curve type: Start: 0: real geometry 1: construction geometry Line: ending trim parameter (for filleting) Ellipse: not applicable Elliptical arc: not applicable

		Bezier spline: not applicable Fillet: not applicable

REFGEO - Attachment Point

Name	Data	Description
X	L ARRAY	Translation along x-axis
Y	L ARRAY	Translation along y-axis
Z	L ARRAY	Translation along z-axis
RotX	T ARRAY	X-component of rotation axis
RotY	T ARRAY	Y-component of rotation axis
RotZ	T ARRAY	Z-component of rotation axis
RotAngle	A ARRAY	Rotation angle

SOLID - All the solid-based shape components, including: Sweep - extrude, spin, general sweep, Loft, Brep, and Compound. Each variable below is marked with the shape components that may own it using the following scheme. Most variables can be owned by more than one type of shape components.

A: for all solid-based models
B: for Brep
C: for Compound
E: for Sweep that is linear (extrude)
T: for Sweep that is about an axis (spin)
G: for Sweep that is along a general path (general sweep)
L: for Loft

Name	Data	Description
OperationType	N[0]	**(A)** Operation type 0: unite 1: subtract

		2: intersect 3: blend 4: chamfer 5: no operation
BlendAll	L	**(A)** Blend radius for all the edges
BlendAllInters	L	**(A)** Blend radius for all the intersection edges
ChamferAllLeft	L	**(A)** Chamfer depth on the left side of all the edges
ChamferAllRight	L	**(A)** Chamfer depth on the right side of all the edges
ChamferAllInter sLeft	L	**(A)** Chamfer depth on the left side of all the intersection edges
ChamferAllInter sRight	L	**(A)** Chamfer depth on the right side of all the intersection edges
CapHeight	L	**(ETGL)** Height of an end cap
ShellType	N(0-4) [0]	**(ETGL)** Shelling type of a shelled shape 0: no shelling 1: open at both ends 2: open at bottom 3: open at top 4: closed at both ends
ShellThickness	L	**(ETGL)** Wall thickness of a shelled shape
ShellTopOffset	L	**(ETGL)** For multiple-shape shelling, height adjustment at the top of an extruded shape typically used to connect its shell with the shell of its parent shape
ShellBotOffset	L	**(ETGL)** For multiple-shape shelling, height adjustment at the top of an extruded shape typically used to connect its shell with the shell of its parent shape

BlendSide	L	**(ETGL)** Blend radius for side edges
BlendTop	L	**(ETGL)** Blend radius for top edges
BlendBot	L	**(ETGL)** Blend radius for bottom edges
BlendShell	L	**(ETGL)** Blend radius for the edges of the shell
ChamferSideLeft	L	**(ETGL)** Chamfer depth on the left side of the side edges
ChamferSideRight	L	**(ETGL)** Chamfer depth on the right side of side edges
ChamferTopLeft	L	**(ETGL)** Chamfer depth on the left side of the for top edges
ChamferTopRight	L	**(ETGL)** Chamfer depth on the right side of top edges
ChamferBotLeft	L	**(ETGL)** Chamfer depth on the left side of the bottom edges
ChamferBotRight	L	**(ETGL)** Chamfer depth on the right side of bottom edges
ChamferShellLeft	L	**(ETGL)** Chamfer depth on the left side of the shell’s edges
ChamferShellRight	L	**(ETGL)** Chamfer depth on the right side of shell’s edges
MatchTop	N(0-1) [0]	**(ETGL)** For a shape dropped on another one 0: no need to match top face with face of parent shape 1: match top face with face of parent shape
MatchBot	N(0-1) [0]	**(ETGL)** For a shape dropped on another one 0: no need to match bottom face with face of parent shape 1: match bottom face with face of parent shape
MatchTopShell	N	**(ETGL)** For multiple-shape shelling, extends the top of the shell to be flush

		with the face of the parent shape
MatchBotShell		**(ETGL)** For multiple-shape shelling, extends the bottom of the shell to be flush with the face of the parent shape
MatchTopBot	N[0]	**(E)** For an extruded shape dropped on another shape, if a face is matched to the face of the parent shape, this variable can make the other face to use the same surface geometry of the parent face. 0: do not match the parent's surface geometry 1: match the parent's surface geometry
Angle	A[2PI]	**(T)** Turn angle of a turn shape
ShellAxial	N(0-1) [0]	**(T)** This applies to shelling of a turn shape 0: do not shell along spin axis 1: shell along spin axis axis
DraftPlaneTop	A	**(ETG)** Draft angle for tapering at the top
DraftPlaneOrien tTop	A	**(ETG)** Index angle for tapering at the top; it indicates the orientation of the draft plane (taper)
DraftPlaneSide	A	**(ETG)** Draft angle for tapering along sweep direction
XsectAlign	N(0-1) [0]	**(L)** Alignment of loft cross sections with respect to path 0: a cross section is always perpendicular to the path 1: all cross sections are parallel to one another
FacetApproxLev el	T[24]	**(C)** Number of polygons used to represent a short cylindrical face

EDGEBLCH - blend or chamfer for an edge

Name	Data	Description
OperationType	N(3-4)	3: blend edge, 4: chamfer edge
Radius1	L	For constant-radius blend - blend radius For variable-radius blend - first blend radius For chamfer - depth on the left side of the edge
Radius2	L	For variable-radius blend - second blend radius For chamfer - depth on the right side of the edge
BevelType	N	Edge blend or chamfer type 1: constant radius blend 2: variable radius blend 3: not used 4: chamfer

ROTATOR - triball location

Name	Data	Description
X	L[0]	triball location along length
Y	L[0]	triball location along width
Z	L[0]	triball location along height

TEXT - formats and displays text with variable font types and sizes

Name	Data	Description
TextStyle	N(1-5) [1]	Specifies the style of the text 1: 2D text, text has no depth 2: 3D text, sharp edges

		3: 3D text, edges are beveled (chamfered) 4: 3D text, edges have smooth bevel (rounded) 5: 3D text, edges have cornice bevel (reversed round)
TextDepth	L[1]	Depth of text
BevelScale	T[1]	A factor that determines the steepness of the bevel; the smaller the steeper
BevelDepth	L[1]	The amount of bevel setback along the depth
BevelNumFaces	N[2]	The number of divisions used to represent a smooth bevel
Content	S	The text to be drawn
FontName	S[Arial]	The default fontname
FontSize	N[720]	The default fontsize in twips (.05 point)
FontStyle	N(0-1) [0]	The default font style options in hexadecimal 0x1: bold 0x2: italic
TextHAlign	N(0-2) [0]	The default horizontal text alignment 0: left 1: center 2: right
TextVAlign	N(0-3) [1]	The default vertical text alignment 0: bottom 1: center 2: top
Font2Page	T[254/144]	The scaling from font units to page units. Font units are almost always in twips (.05 points). Page units are himetric (.01 mm). The default

		scaling is correct for twips to himetric.
Page2Model	T[0.1]	The scaling from page units to model units.
EditBoxExtent	N ARRAY	The width and height of the edit control, in page units.
EditBoxPosition	N ARRAY [0,0,0]	The position of the edit control, relative to the anchor position of the shape, in page units.
PreScaleTranslate	L ARRAY [0,0,0]	A translation on text facets before final scaling
Scale	T ARRAY [1,1,1]	A final scaling on text facets before rendering
PostScaleTranslate	L ARRAY [0,0,0]	A translation on text facets after final scaling

UIBEHAVIOR - controls the behavior when a shape is dragged, dropped, or double-clicked

Name	Data	Description
MoveControl	N(1-3)[3]	Controls how an object is moved when dragged 1: no movement 2: move on surface 3: move in space
DropEvent	N(1-4)[1]	Controls what happens when an object is dropped 1: current default behavior 2: edit properties 3: edit text 4: edit embedded objects
DoubleClickEvent	N(1-4)[1]	Controls what happens when an object is double-clicked

		1: current default behavior 2: edit properties 3: edit text 4: edit embedded objects

LIGHT - light for illumination

Name	Data	Description
On	N(0-1)[1]	0: light off 1: light on
ShadowType	N(0-1)[0]	0: does not cast shadows 1: casts shadows using shadow buffer
Intensity	T[1]	Light intensity
Red	T(0-1)[1]	Red component of light
Green	T(0-1)[1]	Green component of light
Blue	T(0-1)[1]	Blue component of light
SpotConcentration	T[0]	Exponent of the cosine variation of concentration
SpotConeAngle	A[PI/2]	Angle at which the light begins to fall off
SpotFallOffAngle	A[0]	Measured from the fall-off angle, the angle interval over which the light falls to zero linearly
ShadowFuzz	N(0-1)[1]	Controls sharpness of shadows 0: sharp shadow edges 1: fuzzy shadow edges
ShadowResolution	N(10-1000)[256]	Controls the quality of shadows, higher values means better quality but also more memory consumption

LSECTION - positioning of a profile used as a cross-section for loft

Name	Data	Description
PositionOnCurve	T(0-1)	Parameter value of the point on the path where the section crosses
SpinAngle	A	Spin angle (twist) about the tangent at the

		path

DIMENSION

Name	Data	Description
Type	N(1-3)	The type of dimension 1: linear dimension 2: angular dimension 3: radial dimension
X	L ARRAY	X-coordinate of start and end points
Y	L ARRAY	Y-coordinate of start and end points
Z	L ARRAY	Z-coordinate of start and end points
NormalX	T ARRAY	X-component of normal of end point (if end point is not a plane, normal is 0,0,0)
NormalY	T ARRAY	Y-component of normal of end point (if end point is not a plane, normal is 0,0,0)
NormalZ	T ARRAY	Z-component of normal of end point (if end point is not a plane, normal is 0,0,0)
Locked	N(0-1)[0]	Controls if a dimension is a constraint 1: use dimension as a constraint during cruising, locked against changes when solving smart dimensions 0: otherwise
Prefix	S	Prefix text string for dimension
Postfix	S	Postfix text string for dimension
ExtensionDistance	L	Distance of the extension lines
ExtensionDirX	T	X-component of direction of extension lines (for linear dimension only)
ExtensionDirY	T	Y-component of direction of extension lines (for linear dimension only)
ExtensionDirZ	T	Z-component of direction of extension

		lines (for linear dimension only)
Value	T	Latest measured value of dimension (not to be set by user)
HandleFixedPt	N(1-2)[2]	Specifies stationary point when custom handles are dragged 1: mid-point between the two opposite handles 2: the opposite handle
HandleDisplay Type	N(0-3)[0]	Specifies which handle is displayed 0: both handles 1: first handle 2: second handle 3: no display

VARIABLE - data of an array of variable values and names

Name	Data	Description
Number	T ARRAY	Typically the value of a variable
Name	S ARRAY	Typically the name of the variable

EMBEDDEDPAGE - embedded page

Name	Data	Description
CamEyeX	L[500]	X-coordinate of eye point of camera
CamEyeY	L[490]	Y-coordinate of eye point of camera
CamEyeZ	L[480]	Z-coordinate of eye point of camera
CamEyeUpX	T[-0.4039]	X-component of the “up” direction of camera
CamEyeUpY	T[-0.3958]	Y-component of the “up” direction of camera

CamEyeUpZ	T[0.8248]	Z-component of the “up” direction of camera
CamToX	L[0]	X-coordinate of the point at which the camera is aiming
CamToY	L[0]	Y-coordinate of the point at which the camera is aiming
CamToZ	L[0]	Z-coordinate of the point at which the camera is aiming
CamProjection	N[0]	Viewing projection 0: perspective 1: parallel (orthographic)
CamAngle	A[0.5236]	Angular field of view
CamScale	T[1]	For orthographic viewing with true lengths, the drawing scale (e.g. 4 means 4 times the true length)
Transparent	N(0-1)[0]	Controls if the background is displayed 0: background is opaque 1: background is transparent
RenderStyle	N[7]	Rendering style 0: no rendering 1: draw in wireframe 2: draw in wireframe using z-buffer for lines 3: no rendering - use z-only for shadow buffers 4: no shading, fills facet with diffuse color 5: constant facet shading 6: standard gouraud shading 7: color textures and reflection maps 8: per-pixel shading, can have options defined by or’ing one or more of the following hexidecimal values

		0x10: antialias 0x20: cast shadows 0x40: ray-trace 0x100000: if this bit is set, only a layout (2D drawing) will be drawn.
Accuracy	N[1000]	For layouts, controls how accurate curves are discretized for displaying, for example 500: coarse 1000: normal 3000: fine
VisibleSilhStyle	N(0-3)[0]	For layouts, line style for visible silhouette lines 0: solid 1: invisible 2: dashed 3: dotted
InvisibleSilhStyle	N(0-3)[3]	For layouts, line style for hidden silhouette lines 0: solid 1: invisible 2: dashed 3: dotted
VisibleStyle	N(0-3)[0]	For layouts, line style for visible lines 0: solid 1: invisible 2: dashed 3: dotted
InvisibleStyle	N(0-3)[3]	For layouts, line style for hidden silhouette lines 0: solid 1: invisible

		2: dashed 3: dotted
BorderStyle	N(0-3)[1]	For layouts, line style for the border 0: solid 1: invisible 2: dashed 3: dotted
ScaleBehavior	N(0-1)[0]	For layout, controls how the geometry is scaled 0: fit to sizing box 1: preserve original lengths
DisplayDimensions	N(0-1)[1]	Controls whether dimensions are displayed 0: do not display dimensions 1: display dimensions
DisplayModelEdges	N(0-1)[0]	Controls whether model edges are displayed 0: do not display model edges 1: display model edges
DimensionTextSize	L	Specifies text height of dimensions